## Claim Amendments

Please amend the claims to be as follows.

Claims 1-2. (canceled)

3. (currently amended) ~~The method of claim 1, further comprising:~~ A method of obtaining status information from user threads of a target process, the method comprising:

performing a system call from a querying process;

creating a kernel debug thread in a kernel entity of the target process;

creating a user status thread in a user entity of the target process; and

~~the user status thread~~ collecting status information by the user status thread from other user threads of the target process.

4. (original) The method of claim 3, further comprising:

passing the collected status information from the user status thread to the kernel debug thread.

5. (original) The method of claim 4, wherein the user status thread goes into a sleep state after passing the collected status information to the kernel debug thread.

6. (original) The method of claim 4, further comprising:

relaying the collected status information from the kernel debug thread to the querying process.

7. (original) The method of claim 6, wherein the kernel debug thread goes into a sleep state after relaying the collected status information to the querying process.

8. (currently amended) The method of ~~claim 1~~ claim 3, wherein the method is performed in conjunction with an operating system having an MxN thread model.

9. (currently amended) The method of ~~claim 1~~ claim 3, wherein the kernel debug thread is created by the system call.

10. (currently amended) The method of ~~claim 1~~ claim 3, wherein the user status thread is first created in the kernel entity as a scheduler activation and then comes up to the user entity via an upcall handler.

11. (currently amended) The method of ~~claim 1~~ claim 3, wherein the status information does not relate to registers of the user threads, and wherein none of the user threads need to be halted.

12. (currently amended) The method of ~~claim 1~~ claim 3, wherein the status information relates to at least one register of a user thread, and wherein the user thread is temporarily suspended.

13. (currently amended) [[An]] A computer apparatus having an operating system with capability to obtain status information from user threads of a target process so as to debug the target process without requiring stopping the target process, the ~~operating system~~ computer apparatus comprising:

a processor configured to execute program instructions;

memory configured to store program instructions and data; and

processor-readable instructions for a first system call configured to create a kernel debug thread in a kernel entity of the target process; and

processor-readable instructions for a second system call configured to awake the kernel debug thread and pass information to the kernel debug thread.

14. (currently amended) The ~~operating system~~ computer apparatus of claim 13, further comprising:

processor-readable instructions for a thread library configured to register a user status thread's stack and data memory.

15. (currently amended) The ~~operating system~~ computer apparatus of claim 14, further comprising:

processor-readable instructions for a body function in a thread library which is configured to sleep until awaken to act upon a user thread status inquiry.

16. (new) The method of claim 3, wherein the method does not require stopping of the target process.